

Subject: Workshop on Software Reliability

On January 24, 2003 a workshop on software reliability was held at Stevens Institute of Technology. The objective of the meeting was to determine areas of potential collaboration and to scope the state-of-the-art and the state-of-the-practice of software reliability engineering.

Attending:

Name	Phone	E-mail	Affiliation
Bernstein, Larry	973-258-9213	lbernstein@ieee.org	Stevens
Ding, Hui	217-333-3722	huiding@uiuc.edu	Univ. of Illinois
Duggan, Dominic	201-216-8042	Dduggan @cs.stevens-tech.edu	Stevens
Edwards, Rand	908-582-7027	randedwards@lucent.com	Lucent
Ginsberg, Allen	973-408-3198	aginsberg@drew.edu	Drew
Kintala, Chandra	908-696-5400	cmk@avaya.com	Avaya
Klappholz, David	908-464-0805	d.klappholz@att.net	Stevens
Kowshik	217-384-0899	kowshik@uiuc.edu	Univ. of Illinois
Laird, Linda	908-803-0288	Linda_m_Laird@ecomcast.net	Nomadic Storage
Levendel, Haim	312-862-1271	Levendel@sprynet.com	Motorola
Musa, John	973-267-5284	j.musa@ieee.org	Consultant
Sha, Lui	217-244-1887	lrs@cs.uiuc.edu	Univ. of Illinois

Baseline:

1. Libraries available to make software more reliable:
 - a. SWIFT- Provides for recovery and rejuvenation- see Duke
 - b. Libesafe for buffer overflows prevention
2. Safety
 - a. Prevent Memory Leaks
 - b. Inject type 'safety' information in object code
 - c. Contain buffer overflows or overruns
 - d. Static checking before and after upgrades.
 - e. Hot Swapping
 - f. Boundary Checking
 - g. Error control-filter – too often too many alerts are issued and then ignored.
 - h. This often defeated the advantages of Lint for C code.
 - i. FCC switch reliability data as a source for analysis
 - j. Lost memory recovery and defragmentation
3. Bio-medical reach out
 - a. networking of instruments
 - b. Embedded software
4. Model checking and model-based development.

5. Availability not just reliability is the issue
6. Capers' Jones rule: Testing is 30% effective in finding bugs, yet inspections are 70% effective. (Beware bureaucratic inspection processes force out the best software engineers. The best are often 20 times more effective than most. Only 1-2% of the software population is in this best class, yet 80% of software engineers think they are best.)
7. "Requirements inspection, design inspection and code inspection each reduce bugs by 50%. The order of cost (least to most) is Requirements Inspections, then Design Inspections, then Code Inspections," stated John Musa
8. Application of Control System Theory to Software Design by Sha.
9. Musa's world class course in Software Reliability with an emphasis on operational profiles.
10. Software Black Holes by Levendel- 'Bugs abhor Loneliness'
11. Most successful managers only track modification or failure reports

Consensus Points:

1. Just right reliability is needed.
2. Reliability must be engineered
3. Failure Detection and Backup standard libraries should be widespread
4. System of Systems reliability needs to be engineered
5. Patches need to be made robust so that known fixes that prevent virus attacks will be installed.
6. Feedback Control needs to be leveraged (and taught).
7. Deviations are departures from expected behavior. Fault tolerance detects deviations and prevents them from becoming failure.
8. Failures are crashes or hangs
9. Failures are faults that get executed or result from incorrect data input/output or from hardware problems. Further, crashes may be easily recovered from and not be a problem and hence not a failure to users in some cases. Approaches that prevent faults from being executed need to be propagated and additional ones invented. Approaches to increase the availability of systems by decreasing the re-launch time after a failure are also needed.

Potential Collaborations:

1. Handbook of Software Reliability – (Laird and Ginsberg)
 - a. Case studies
 - b. Practices with qualified domains of application
 - c. Expert System assists to find applicable practice
 - d. Cost/benefit ratios specified for each process
 - e. Complexity/effort ratios specified for each process
 - f. Architecture patterns suitable for application degraded modes.
2. Establish a recommended reading list for Software Engineers:

- a Multics Paper by Corboto
 - b. Spiral model by Boehm
 - c. Gotos are Harmful by Dijkstra
 - d. There is no Silver Bullet by Brooks
 - e. Mythical Man Month by Brooks
 - f. Peopleware
 - g. Software Project Management audits by Bernstein
 - h. Software Simplicity by Sha
 - i. Software Reliability by Musa
 - j. Modularization by Parnas
 - k. Software Risks by Newman
 - l. Bentley
- modification/failure reports

3. Software Reliability Research on Wireless ARQ: (Bernstein and Sha)
 - a. Algorithm robustness
 - b. Software Processes
 - c. Software Libraries that increase reliability
 - d. Extensions to Architecture description languages
4. VoIP (Levendel, Kintala, Sha, Edwards)
 - a. Model clash between datagrams and latency
 - b. Router QoS controls
 - c. Networks and System pacing
5. Reach out to new industries beyond telecommunications- (Musa and Bernstein)
6. Hot Swapping (Duggan and Ding)
7. Virus protection to homes.